SEGA™

SEGA OF AMERICA, INC.
Consumer Products Division

32X
Hardware Manual Supplement 2

Doc. # MAR-32-R4-SP2-072694

32X Hardware Manual Supplement 2

Limitations Concerning the SH2 Interrupt

Poor perfomance occurs in the SH2 concerning the following interrupts.

1. If an external interrupt (VRES, V, H, CMD, PWM) input is input in the acknowledge period for interrupt inputs, or external interrupt of lower levels, SH2 will not recognize the external interrupt.
2. When multiple interrupt inputs are entered, there may be branching to the interrupt process routine of a vector number that differs from the interrupt vector originally received. Nevertheless, an accurate value is entered in the SR mask level.

Corrective Action

1. Corrective action is taken by controlling the free-run-timer output of SH2 by software. The corrective process must be done within the external interrupt process routine. A pipeline operation must be considered to prevent the same interrupt from being duplicated.
2. The jump destination of all interrupts, internal and external, are set to the same address and can be avoided by jumping to the original jump destination through theSR value.

Precautions

a) The SR mask should be set to level 1; normal operation will not occur if set at 0.
b) Interrupt of the SH2 internal peripheral module should use levels 2 ~ 5.
c) With the EVA chip cut 2.5, operation is normal although no corrective action is taken since the trouble above is corrected, but because an unmodified chip is used in the initial version of the actual device, corrective action must be taken.

When clearing the external interrupt factor by the program, the pipeline operation must be considered in order that the same interrupt is not applied again. When interrupt factor clear is written to the I/O address, the next instruction is executed before the write operation is completed through the effect of the write buffer. In order to execute the next command after completing the write operation, and if write continues and read is performed from the same address, synchronization is completely done.

As Figure 1 shows, when returning from the interrupt process through RTE, a 1 cycle interval is required between the read command for synchronization and the RTE command. When changing SR value through the LDC command and allowing other interrupts to apply in multiples, a minimum 4 cycle interval is required in between synchronous command and LDC command, as shown in Figure 2.

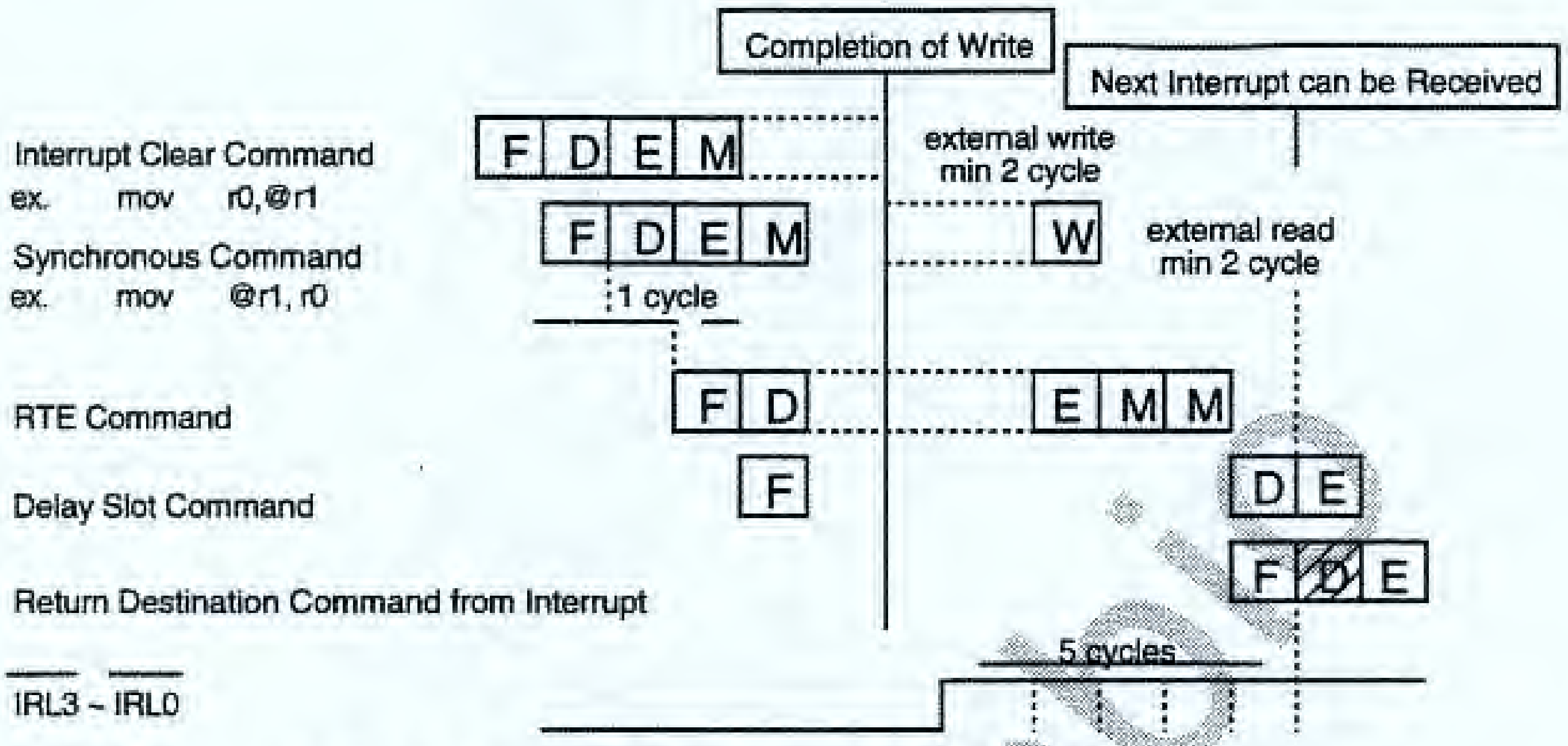

Figure 1 Pipeline Operation When Returning by RTE

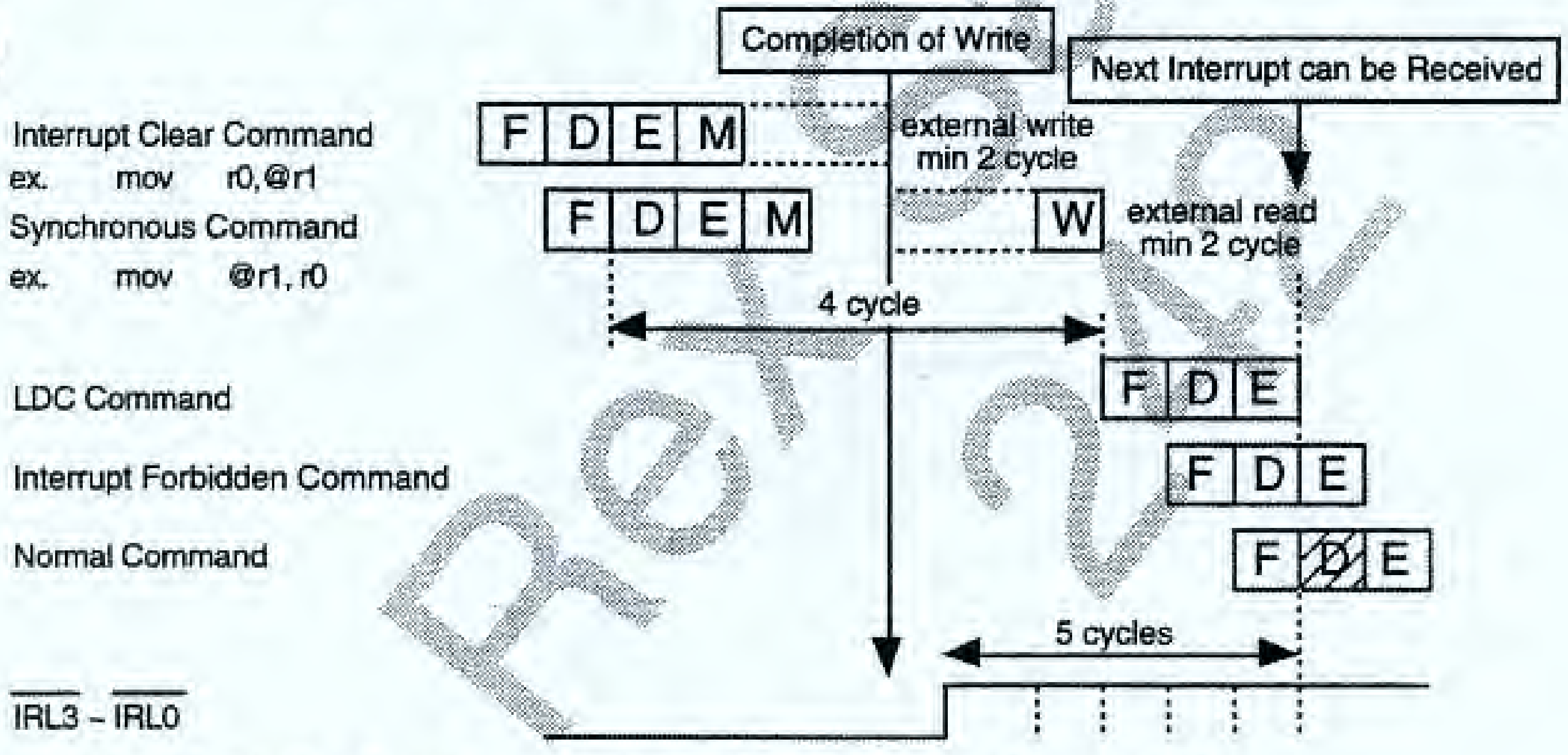

Figure 2 Pipeline Operation when Authorizing Interrupt by Change of SR

The pipeline operation must be considered in keeping the same interrupt from reoccurring (reapplying) when the interrupt factor is from the internal peripheral module. Two cycles are needed until the interrupt from the internal peripheral module is recognized by the CPU, and to transmit interrupt requests that no longer exist. When returning from the interrupt process through RTE, as shown in Figure 3, there is a 1 cycle margin until interrupt is received, even if he RTE command is executed immediately after the read command for synchronization. When authorizing the change of the SR value through the LDC command and other multiple interrupts, a minimum 2 cycle interval is required in between synchronous command and LDC command, as shown in Figure 4.

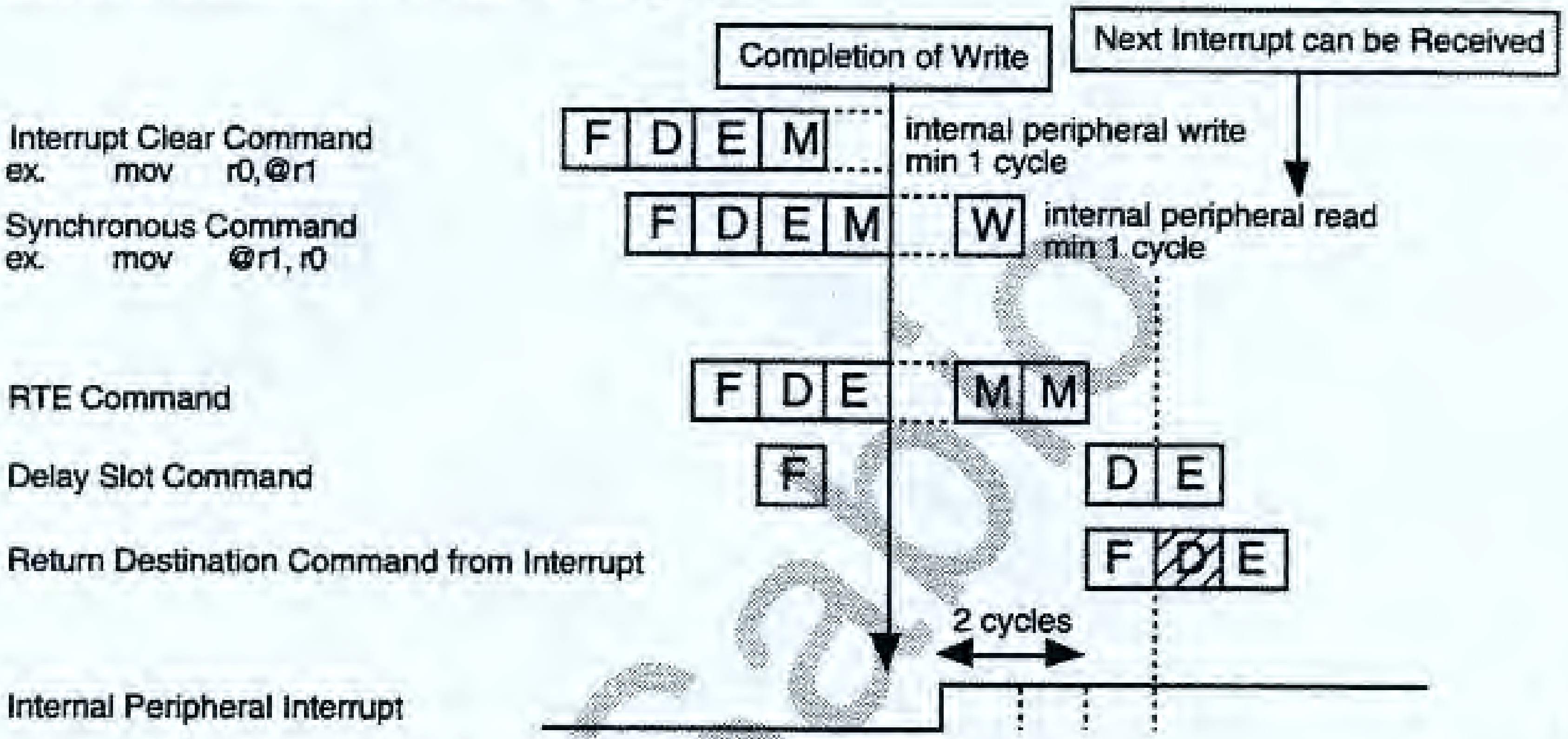

Figure 3 Pipeline Operation When Returning by RTE

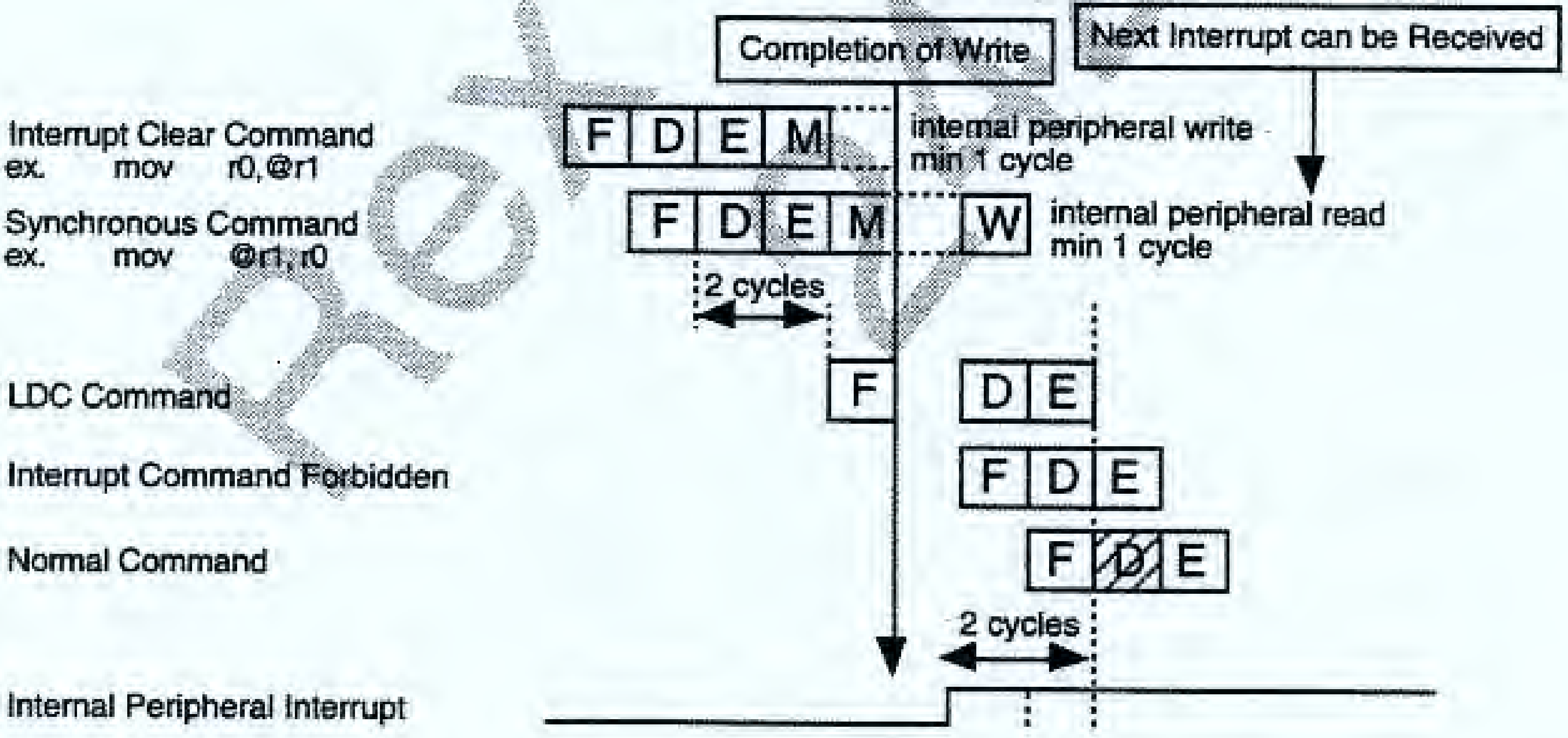

Figure 4 Pipeline Operation when Authorizing Interrupt by Change of SR

```
;---------------------------------------------------------------------------------------------------
;        Interrupt Correction Sample Program (July 6, 1994)
;---------------------------------------------------------------------------------------------------
CS0                 .equu   h'00000000        ;  Boot ROM, Register
CS1                 .equ    h'02000000        ;  Cartirdge ROM
CS2                 .equ    h'04000000        ;  Frame buffer
CS3                 .equ    h'06000000        ;  SDRAM

TH                  .equ    h'20000000        ;  Cache through
CS0TH               .equ    h'20000000        ;  Boot ROM, Register (Cache through)
CS1TH               .equ    h'22000000        ;  Cartridge ROM (Cache through)
CS2TH               .equ    h'24000000        ;  Frame Buffer (Cache through)
CS3TH               .equ    h'26000000        ;  SDRAM (Cache through)

_SERIALMODE         .equ    h'fffffe00        ;  Serial Mode Register

_FRT                .equ    h'fffffe10        ;  Free Run Timer
_TIRE               .equ    h'00              ;  Timer Interrup Enable Register
_TCSR               .equ    h'01              ;  Timer Control & Status Register
_FRC_H              .equ    h'02              ;  Free Running Counter High
_FRC_L              .equ    h'03              ;  Free Running Counter Low
_OCR_H              .equ    h'04              ;  Output Compare Register High
_OCR_L              .equ    h'05              ;  Output Compare Register Low
_TCR                .equ    h'06              ;  Timer Control Register
_TOCR               .equ    h'07              ;  Timer Output Compare Control Register

_CCRREG             .equ    h'fffffe92        ;  Cache Control Register

_JR                 .equ    h'ffffff00        ;  DIVU
_HRL32              .equ    h'ffffff04        ;  DIVU
_HRH                .equ    h'ffffff10        ;  DIVU
_HRL                .equ    h'ffffff14        ;  DIVU

_DMASOURCE0         .equ    h'ffffff80        ;  DMA  Source Address 0
_DMADEST0           .equ    h'ffffff84        ;  DMA  Destination Address 0
_DMACOUNT0          .equ    h'ffffff88        ;  DMA  Transfer Count 0
_DMACHANNEL0        .equ    h'ffffff8c        ;  DMA  Channel Control 0
_DMASOURCE1         .equ    h'ffffff90        ;  DMA  Source Address 1
_DMADEST1           .equ    h'ffffff94        ;  DMA  Destination Address 1
_DMACOUNT1          .equ    h'ffffff98        ;  DMA  Transfer Count 1
_DMACHANNEL1        .equ    h'ffffff9c        ;  DMA  Channel Control 1
_DMAVECTORN0        .equ    h'ffffffa0        ;  DMA  Vector No. N0
_DMAVECTORE0        .equ    h'ffffffa4        ;  DMA  Vector No. E0
_DMAVECTORN1        .equ    h'ffffffa8        ;  DMA  Vector No. N1
_DMAVECTORE1        .equ    h'ffffffac        ;  DMA  Vector No. E1
_DMAOPERATION       .equ    h'ffffffb0        ;  DMA  Operation
_DMAREQACK0         .equ    h'ffffffb4        ;  DMA  Request / Ack Select Control 0
_DMAREQACK1         .equ    h'ffffffb8        ;  DMA  Request / Ack Select Control 1

;
;        SYSREG
;
_sysreg             .equ    h'00004000+TH     ;  SYSREG
adapter             .equ    h'00              ;  Adapter Control Register
intmask             .equ    h'01              ;  Interrupt Mask
standby             .equ    h'02              ;  Standby Mode Shift
hcount              .equ    h'05              ;  H Interrupt Counter Reister {Note: Typo may be in code}
vdpfifo             .equ    h'06              ;  Frame Buffer FIFO Condition
dreqctl             .equ    h'07              ;  DREQ Control
dreqsource          .equ    h'08              ;  DREQ Source Address
dreqdest            .equ    h'0c              ;  DREQ Destination Address
```

```
dreqlen             .equ   h'10                  ; DREQ Length
fifo                .equ   h'12                  ; FIFO
vresintclr          .equ   h'14                  ; VRES Interrupt Clear
vintclr             .equ   h'16                  ; V Interrupt Clear
hintclr             .equ   h'18                  ; H Interrupt Clear
cmdintclr           .equ   h'1a                  ; CMD Interrupt Clear
pwmintclr           .equ   h'1c                  ; PWM Interrupt Clear
comm0               .equ   h'20                  ; Communication Port
comm2               .equ   h'22                  ;
comm4               .equ   h'24                  ;
comm6               .equ   h'26                  ;
comm8               .equ   h'28                  ;
comm9               .equ   h'29                  ;
comm10              .equ   h'2a                  ;
comm12              .equ   h'2c                  ;
comm14              .equ   h'2e                  ; PWM Timer Control
timerctl            .equ   h'30                  ; PWM Control
pwmctl              .equ   h'31                  ; PWM
cycle               .equ   h'32                  ;
lchwidth            .equ   h'34                  ;
rchwidth            .equ   h'36                  ;
monowidth           .equ   h'38                  ;

;
;       VDPREG.
;
_vdpreg             .equ   h'00004100+TH         ; VDPREG.
tvmode              .equ   h'00                  ; TV Mode Register
bitmapmd            .equ   h'01                  ; Bitmap Mode Register
shift               .equ   h'03                  ; Shift Control Register
filllength          .equ   h'05                  ; Auto Fill Length Register
fillstart           .equ   h'06                  ; Auto Fill Start Address Register
filldata            .equ   h'08                  ; Auto Fill Data Register
vdpsts              .equ   h'0a                  ; VDP Status Register
framectl            .equ   h'0b                  ; Frame Buffer Control Register

_palette            .equ   h'00004200+TH         ; Palette RAM
_framebuffer        .equ   CS2TH                 ; Frame Buffer
_overwrite          .equ   CS2TH+h'20000         ; Over Write Image

;
;       SH2 Vector
;
vector:
        .data.l     start                        ; Power On Reset PC
_stack:
        .data.l     CS3+h'3ff00,                 ; Power On Reset SP
+                   start                        ; Manual Reset PC
+                   CS3+h'3ff00                  ; Manaual Reset SP
        .data.l     error0,                      ; General Invalid Command
+                   h'00000000                   ; System Reserve
+                   eror0,{Note:typo may be in code} ; Slot Invalid Command
+                   h'20100400,                  ; System Reserve (ICE Vector)
+                   h'20100420,                  ; System Reserve (ICE Vector)
+                   error0,                      ; CPU Address Error
+                   error0,                      ; DMA Address Error
+                   error0,                      ; NMI
+                   error0,                      ; User Break
        .datab.l    19, h'00000000               ; System Reserve
        .datab.l    32, error0                   ; Trap Command
        .data.l     m_int,                       ; Interrupt 1
```

```
+                       m_int,                  ; Interrupt 2, 3
+                       m_int,                  ; Interrupt 4, 5
+                       m_int,                  ; Interrupt 6, 7
+                       m_int,                  ; Interrupt 8, 9
+                       m_int,                  ; Interrupt 10, 11
+                       m_int,                  ; Interrupt 12, 13
+                       m_int,                  ; Interrupt 14, 15

;
;       Program Start
;
Start:
                mov.l   #_sysreg, r14
                lcd     r14, gbr

                mov.l   #_FRT, r1               ; Set Free Run Timer
                mov     #h' 00, r0
                mov.b   r0, @ (_TIER, r1)
                mov     #h' e2, r0
                mov.b   r0, @ (_TOCR, r1)
                mov     #h' 00, r0
                mov.b   r0, @ (_OCR_H, r1)
                mov     #h' 01, r0
                mov.b   r0, @ (_OCR_L, r1)
                mov     #0, r0
                mov.b   r0, @ (_TCR, r1)
                mov     #1, r0
                mov.b   r0, @ (_TCSR, r1)
                mov     #h' 00, r0
                mov.b   r0, @ (_FRC_H, r1)
                mov.b   r0, @ (_FRC_H, r1)
wait md:
                mov.l   @ (comm0, gbr) , r0     ; Timing with Mega Drive
                cmp/eq  #0, r0
                bf      wait_md

                mov     #h'20, r0
                ldc     r0, sr                  ; SH2 Interrupt Enable

                        |
                        |
;
;       Interrupt Control
;
m_int:
                push    0, 1
                sts.l   pr, @-r15

                stc     sr, r0
                shlr2   r0
                and     #h'3c, r0
                mov.l   #inttable, rl
                add     r1, r0
                mov.l   @r0, r1
                jsr     @r1
                nop

                lds.l   @r15+, pr
                pop     0,1
                rte
                nop
```

```
                .align      4
inttable:
        .data.l     noret,                                  ; Illegal Interrupt
+                   noret, noret, noret, noret, noret,      ; Level 1 _ 5
+                   pwmint, pwmint, cmdint, cmdint          ; Level 6 _ 9
+                   hint, hint, vint, vint, vresint, vresint ; Level 10 _ 15
;       Odd and even levels for external interrupt vectors should be the same address, as above.

;
;       Ignore
;
noret:
                rts
                nop

;
;       VRES Interrupt
;
vresint:
                mov.l       #sysreg, r0
                ldc         r0, gbr

                mov.w  r0, @ (vresintclr, gbr)      ; V Interrupt Clear

                mov.l       #_stack,r1              ; Stack Ponter Change
                mov.l       @r1, r15

                mov.l       #_hotstart, r0
                mov         r0, @r15                ; PC Change
                mov.w       #h'f0, r0
                mov         r0, @ (4, r15)          ; SR Mask

                rte
                nop

;
;       V  Interrupt
;
vint:
                stc.l       gbr, @-r15

                mov.l       #_sysreg, r0
                ldc         r0, gbr

                mov.l       #h'f0, r0               ; Interrupt Mask
                ldc         r0, sr

                mov.l       #_FRT, r1               ; External Interrupt Corrective Action
                mov.b       @ (_TORC, r1), r0
                xor         #h'02, r0
                mov.b       r0, @ (_TORC, r1)

                mov.w       r0, @ (vintclr, gbr)    ; V Interrupt clear
;       Other processes (5 clock or more required)
                                |
                                |
                ldc.l           @r15+, gbr
                rts             |
                nop             |
;       The above should be done the same for H, CMD, PWM also.
```